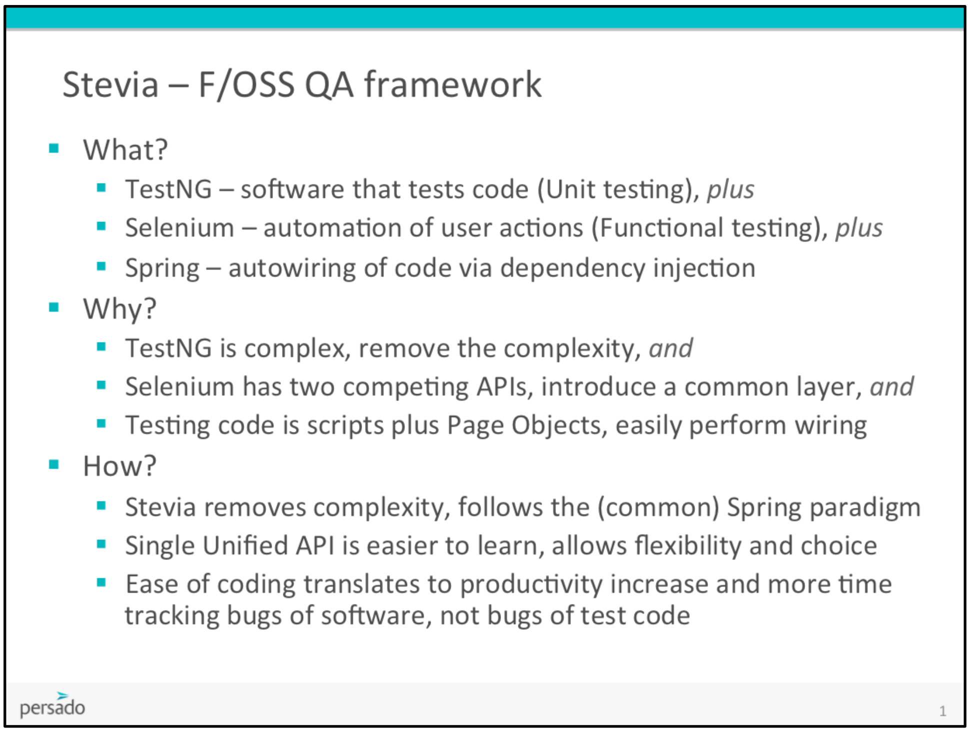

Stevia – F/OSS QA framework

- What?
  - TestNG – software that tests code (Unit testing), *plus*
  - Selenium – automation of user actions (Functional testing), *plus*
  - Spring – autowiring of code via dependency injection
- Why?
  - TestNG is complex, remove the complexity, *and*
  - Selenium has two competing APIs, introduce a common layer, *and*
  - Testing code is scripts plus Page Objects, easily perform wiring
- How?
  - Stevia removes complexity, follows the (common) Spring paradigm
  - Single Unified API is easier to learn, allows flexibility and choice
  - Ease of coding translates to productivity increase and more time tracking bugs of software, not bugs of test code

persado 1

Stevia is the QA framework we created when looking for ways to increase productivity (aka. Scalability) of our QA teams.

It uses the proven recipe of TestNG (a well known unit testing framework) plus Selenium (a well known framework for automation of user actions via browsers) together with a pinch of Spring (the most commonplace user library in Java).

Stevia binds the three components together to create a new approach plus supporting code (hence framework) to automate application testing with focus on simplicity and productivity increase:

Why is this necessary? It is necessary because software becomes more complex by definition and we need to strip down the complexity to be able to perform. We do this via the following strategy for each of the components:

a) TestNG : - a subset is only used, special configuration plus annotations that make the code clear and concise.
b) Selenium :- Webdriver and Selenium-RC have competing APIs that do not follow the same strategy – as if build by different people. This is confusing and hinders adoption of the newer Webdriver – a much more concise and up-to-date implementation and in some cases, faster than Selenium RC.
c) Finally, using Spring is more intuitive and saves time when we want to create links